A1 Cont.

program to view the files is called a web browser ("browser"). A collection of related data files under a common Internet network domain name is commonly referred to as a Web site.

Please replace the paragraph beginning on page 12, line 14, through page 13, line 2, with the following paragraph.

One skilled in the art will easily recognize that URL requests or request header information may be parsed, searched, and read to obtain parameter information being passed to the origin web server. For example, parameters may be obtained from cookies as well as from data information contained in header information. In some cases, the parameters passed A2 are based on the URI or local information, such as the position within the URL address (herein called positional parameters). For example, if the URL request is "http://www.myAuction.com/aw/ listings/list/all/category513/index.html," the parameter, "513," is obtained from "category513." A template, syntax, or pattern match of the URL address may be written to enable the dynamic content cache to extract only certain portions of the URL address.

Please replace the paragraph on page 21, lines 6-13, with the following paragraph.

To determine dependencies, for example, for Figures 6A, 6B, and 6C, a developer has to understand what underlying fields or data sources are used in the dynamic page, and based on those fields or data sources recognize what events invalidate the page. In the preferred A3 embodiment, dependencies are specified with a start tag and closing tag, to be in the form "<Dependency>event-name(parameter_name=parameter_value,...)</Dependency>." A dynamic page may contain a dependency data containing more than one dependency. The event-name may be any user-defined event name; however, event names defined in the dependencies must correspond or relate to those defined in the events (further discussed below).

Please replace the paragraph on page 22, lines 1-13, with the following paragraph.

In accordance with the present invention, the dynamic content cache parses URL A4 addresses, including request header information, and obtains the parameters passed or sent to the origin server. The parameters sent dictate what underlying data are to be displayed.

Referring back to Figure 6A, the parameter CatID with value "131" (i.e., "Playing Cards"), obtained from URL request 602, is passed to the origin server; thus, only items under such category are displayed. The dynamic page 600 is dependent on the underlying data, and any change to the underlying data invalidates the dynamic page. In Table V, for example, line 1 indicates that the dynamic page 600 is dependent on having no status changed (from "open" to "closed") for any item listed under the "131" category; line 2 indicates that it is dependent on having no additional bid be placed on any item listed under the "131" category; line 3 indicates it is dependent on having no change made to the title of any item listed under "131" category; and line 4 indicates that it is dependent on having no additional item added under "131" category. Any events affecting the underlying data invalidate the page.

Please replace the paragraph beginning on page 25, line 19, through page 26, line 3, with the following paragraph.

The EventTemplateList element, lines 13 to 20, shows the pattern, template, or syntax of dependency/event rules recognized by the dynamic content cache, without the actual parameter values. The EventTemplateList acts like a template or a pattern of URL requests that the dynamic content cache monitors. EventTemplate elements, lines 14 to 19, are preferred to be in the syntax of "<EventTemplate>event-name(parameter-name, ...)</EventTemplate>. The event-name is developer-defined. The event-name is preferred to describe the event that invalidates the underlying data in a dynamic page.

Please replace the paragraphs on page 29, lines 3-15, with the following paragraphs.

The EventRuleList element, found in line 46, instructs the dynamic content cache that whenever it receives a URL request with pattern, "http://www.MyAuction.website/createItem.asp," the dynamic content cache is to generate change events (or events) listed in lines 48 and 49. In the preferred embodiment, these events are incorporated in an event message. An event message may contain one or more event rules, is preferably written in XML or HTML and has the format described in Table VIII below.

Table VIII – Event Message	
Ln	Event Message Syntax or Format
1	<EventMessage>
2	<EventGroup webSiteID=URL eventSource=SourceID time=date-time>
3	<EventRule>event-name(parameter-name=value, . . .)</EventRule>
4	<EventRule>event-name(parameter-name=value, . . .)</EventRule>
5	</EventGroup>
6	</EventMessage>

A6 Cont.

Referring to Table VIII, webSiteID (line 2) indicates the root URL, the eventSource indicates from where the event message originated, and time indicates when the event message was generated. The EventRule elements listed in lines 3 and 4 indicate events that would invalidate cached dynamic pages having any dependency identical or matching those events listed. (The event evaluator 1116 of Figure 11, further discussed below, does this function.)

Please replace the paragraph on page 32, lines 3-11, with the following paragraph.

A7

One skilled in the art will also recognize that the dependencies and events may be modified such that additional data are passed to the dynamic content cache. For example, events and/or dependencies may be modified to include the updated information, which particular record, field, or item was changed, the old data and the updated data, and the like. Furthermore, one skilled in the art will recognize that the dependencies and events may be denoted in many ways depending on implementation; for example, the dependency "ItemChange(CatID=131)" may be denoted as a unique identification number, "A11." Moreover, the name of the elements may be altered or replaced (e.g., "NonCacheableObject" may be changed to "EventObject") without affecting the features of the present invention.

Please replace the paragraph on page 37, lines 3-8, with the following paragraph.

A8

The dynamic content cache looks for dependencies within specially delimited comments (lines 1 and 6) as shown in Table X. In the preferred embodiment, the dependencies are formatted, using HTML or XML, as a comment at the end of the generated page. One skilled in the art will also realize that Web pages that are not generated but manually marked-up or coded may also be appended with appropriate dependencies and accordingly be processed and cached by the present invention.

Please replace the paragraph on page 38, lines 3-14, with the following paragraph.

A9

Figure 10A describes the Request-Based event generator 818, which sits in-process with the dynamic content cache and generates events based on URLs that are received and served by the dynamic content cache. In accordance with the invention, similar to a Request-based dependency generator, a configuration file (e.g., listed in Figures 7A and 7B) is needed by the Request-Based event generator 818. In Figure 10A, it is assumed that the configuration file has been defined in the dynamic content cache. The configuration file is read in step 1002 to learn what events are to be generated for each URL pattern. When a URL request is received in step 1004, it is compared to the previously read patterns. The events to be generated are listed under EventRule elements. For example, if the URL received is "http://myAuction.com/ updateItem.asp?ItemID=6113," (see line 52, Figure 7B), the events to be generated are defined in lines 55 and 56. Thus, in this case an event message, e.g., contained Table IX, is generated in step 1006 and then sent to the change event evaluator 816 in step 1008.

Please replace the paragraph on page 39, lines 10-22, with the following paragraph.

A10

The script-based event generator 822 is a script-accessible component that generates events that correspond to URL patterns defined in the configuration file. Alternatively, the script-based dependency generator 822 is preferably a script-accessible component that has methods for recognizing events as the code encounters them, as well as a method that codes the dependencies as a formatted HTML or XML comment at the end of the generated page. The script-based dependency generator components sit in-process with the origin dynamic